Amendments to the Claims

Claims 1 - 2 (canceled)

Claim 3 (previously presented): The method according to Claim 32, wherein the input document is a structured document.

Claim 4 (previously presented): The method according to Claim 3, wherein the structured document is encoded in Extensible Markup Language ("XML").

Claim 5 (previously presented): The method according to Claim 32, wherein the generated output comprises at least one object representation generated from the input document.

Claim 6 (canceled)

Claim 7 (currently amended): The method according to Claim 33, wherein the second ~~syntax level~~ schema definition is requested by specifying a schema name of [[a]] the second schema definition, to which the generated output must adhere.

Claim 8 (currently amended): The method according to Claim 33, wherein the second ~~syntax level~~ schema definition is requested by specifying a schema name of [[a]] the second schema definition, indicating that the second schema definition is to be used by the validating parser when generating the output.

Claim 9 (previously presented): The method according to Claim 8, wherein the schema name is specified, by the application program, as a feature on an invocation of the validating parser.

Claim 10 (canceled)

Claim 11 (currently amended): The method according to Claim 32, wherein the identification of the first ~~syntax level~~ schema definition in the input document comprises a specification, ~~is specified~~ in the syntax of the input document, of the first schema definition.

Claim 12 (currently amended): The method according to Claim [[11]] 32, wherein the identification of the first schema definition in ~~specification in the syntax of~~ the input document uses a schema location construct in the input document.

Claim 13 (currently amended): A computer-implemented method of casting objects, comprising:

providing a validating parser that is adapted for validating whether syntax elements of an input document conform to a first schema definition identified in the input document while generating output objects, from the validated syntax elements of the input document, that conform to a second schema definition dynamically selected by a consuming application of the generated output objects;

using the validating parser for validating whether the syntax elements of [[an]] the input document conform to the first schema definition, wherein:

the first schema definition is an extended schema;[[,]]

using [[a]] the validating parser, responsive to the validating of the syntax elements, for ~~according to a first syntax level while~~ generating the output objects to conform[[,]] to the ~~from the input using the validating parser, according to a~~ second ~~syntax level~~ schema definition, wherein:

the second schema definition is a base schema from which the extended schema was extended, such that the extended schema defines at least one syntax element that is not defined in the base schema; and

the generating further comprises ~~suppressing, by the validating parser~~ not generating any output object for any of the [[,]] at least one ~~of the validated~~ syntax element that is defined in the extended schema but not defined in the base schema ~~elements from the generated output objects~~ in order that the generated output objects will ~~be valid according~~ conform to the second ~~syntax level~~ schema definition; and

providing the generated output objects, by the validating parser, for use by [[an]] the consuming application ~~program~~.

Claims 14 - 19 (canceled)

Claim 20 (currently amended): The method according to Claim 13, wherein:

the ~~first syntax level~~ second schema definition is the base schema;

an intermediate schema definition extends the base schema by adding at least one syntax element not defined in the base schema; and

the first schema definition extends the intermediate schema definition by adding at least one syntax element not defined in the intermediate schema definition; and

the generating further comprises not generating any output object for any of the at least one syntax element that is defined in the intermediate schema but not in the base schema.

~~represents a plurality of extensions to the second syntax level.~~

Claims 21 - 30 (canceled)

Claim 31 (currently amended): A computer-implemented method of providing validation and parsing ~~for clients~~, comprising:

providing a validating parser adapted for validating an input document according to a first schema definition identified in the input document while generating output, from the validated input document, according to a second schema definition dynamically selected by ~~that enables a client~~ a consuming application of the generated output; ~~to dynamically select a syntax abstraction level for use when generating output from the validating parser;~~

~~obtaining an input document to be validated and parsed for the client;~~

validating syntax elements of the input document with the provided validating parser according to the first schema definition, wherein the first schema definition ~~validation is performed according to a first syntax level~~ is an extended schema which specifies a syntax definition to which the syntax elements of the input document are to adhere; and

responsive to the validating of the syntax elements, parsing the validated syntax elements to generate the output for the consuming application according to the second schema definition,

wherein the second schema definition is a base schema from which the extended schema was extended, thereby suppressing at least one of the validated syntax elements when generating the output for the consuming application, ~~from the input document with the provided validating parser, for use by the client,~~ wherein:

~~the generated output has syntax that conforms to the syntax abstraction level that has been dynamically selected by the client;~~

~~the syntax abstraction level is a less-restrictive version of the first syntax level; and~~

~~each of the suppressed~~ syntax elements is valid according to the ~~first syntax level~~ extended schema but is not valid according to the ~~syntax abstraction level~~ base schema.

Claim 32 (currently amended): A computer-implemented method of applying abstraction by a validating parser, comprising:

using, by a validating parser, a first schema definition ~~syntax level~~ for validating syntax elements when parsing syntax of an input document, wherein the first schema definition is identified in the input document; and

omitting, by the validating parser~~, at least one of the validated syntax elements~~ when generating output from the parsed syntax of the input document, each of at least one of the validated ~~wherein each of the omitted~~ syntax elements which is valid according to the first schema definition ~~syntax level~~ but is not valid according to a second schema definition ~~syntax level~~ for which the output is generated, wherein:

the first schema definition is an extended schema; and

the second schema definition is a base schema from which the extended schema is

extended, such that the extended schema defines at least one syntax element that is not defined in the base schema.

Claim 33 (currently amended): The method according to Claim 32, wherein the second ~~syntax level~~ schema definition is dynamically requested, to the validating parser, by an application program for which the output is being generated.